# Gameplay Header:

1. Include Player header
2. Include Board header
3. Gameplay Class
   1. Public data
   2. Initialise starter function
   3. Create getStarterName function, {return starterName}
   4. Initialise final function
   5. Initialise set function
   6. Initialise game function
   7. Initialise startup function
   8. Initialise frequencyTable function
   9. Private data
   10. Create a string called starterName
   11. Create a Player object named joe
   12. Create a Player object named sid
   13. Create a player object named object
   14. Create player pointer called player
   15. Create player pointer called winner
   16. Create player pointer called loser
   17. Create a board object called board

# Board Header:

1. Include player header
2. Board class
   1. Public data
   2. Initialise single function add parameters int aim, and player pointer
   3. Initialise double function add parameters int aim, and player pointer
   4. Initialise treble function add parameters int aim, and player pointer
   5. Initialise bullseye function add parameters player pointer
   6. Create getBoard function (int x) {return board[x]}
   7. Create getRing function (int x, int i) {return ring[x][i]}
   8. Private data
   9. Create an integer board array and input numbers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 25
   10. Create an 2 dimensional integer ring array and input numbers: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 20, 15, 17, 18, 12, 13, 19, 16, 14, 6, 8, 9, 4, 11, 10, 7, 2, 1, 3, 5 18, 17, 19, 13, 20, 10, 16, 11, 12, 15, 14, 5, 6, 9, 2, 8, 3, 4, 7, 1
   11. Create a player class pointer named player

# Player Header:

1. Player class
   1. Public data
   2. Create string type function named getName and make it return name
   3. Create setName function with parameters string name and make it set this name to name
   4. Create int type function named getSuccessrate and make it return successrate
   5. Create setSuccessrate function with parameters int x and make it set successrate to equal x
   6. Create int type function named getPlayerAverage and make it return playerAverage
   7. Create setPlayerAverage function with parameters int x and make it set playerAverage to equal x
   8. Create int type function named getScore and make it return score
   9. Create setScore function with parameters int x and make it set score to equal x
   10. Create addScore function with parameters int x and make it add x to score
   11. Create subtractScore function with parameters int x and make it subtract x to score
   12. Create int type function named getSets and make it return sets
   13. Create setSets function with parameters int x and make it set sets to equal x
   14. Create addSets function with parameters int x and make it add x to sets
   15. Create int type function named getGames and make it return games
   16. Create setGames function with parameters int x and make it set games to equal x
   17. Create addGames function with parameters int x and make it add x to games
   18. Create int type function named getHits and make it return hits
   19. Create setHits function with parameters int x and make it set hits to equal x
   20. Create addHits function with parameters int x and make it add x to hits
   21. Create int type function named getTotalBullseyes and make it return totalBullseyes
   22. Create setTotalBullseyes function with parameters int x and make it set totalBullseyes to equal x
   23. Create addTotalBullseyes function with parameters int x and make it add x to totalBullseyes
   24. Create int type function named getTotalTurns and make it return TotalTurns
   25. Create setTotalTurns function with parameters int x and make it set TotalTurns to equal x
   26. Create addTotalTurns function with parameters int x and make it add x to TotalTurns
   27. Initialise average function
   28. Initialise aim function
   29. Create integer type function named getAimCase and make it return aimCase
   30. Initialise lastHitCheck function with parameters player class pointer named pointer
   31. Create int type function named getLastHitTotal and make it return LastHitTotal
   32. Create setLastHitTotal function with parameters int x and make it set LastHitTotal to equal x
   33. Create addLastHitTotal function with parameters int x and make it add x to LastHitTotal
   34. Create int type function named getLastHit and make it return lastHit
   35. Create setLastHit function with parameters int x and make it set lastHit to equal x
   36. Create bool type function named getLastHitBool and make it return lastHitBool
   37. Create integer type function named getWinsArray and make it return winsArray
   38. Create addWinnsArray function and make it add one to the row selected
   39. Create integer type function named getTotalFinals and make it return totalFinals
   40. Create addTotalFinals function with parameter integer x and make it add x to totalFinals
   41. Private data
   42. Create string variable called name
   43. Create an integer named successrate and set it to 0
   44. Create double variable named playerAverage and set it to 0
   45. Create an integer named score and set it to 501
   46. Create an integer named sets and set it to 0
   47. Create an integer named games and set it to 0
   48. Create integers named lastHit, lastHitTotal, hits, totalbullseyes, totalTurns, aimCase, totalFinals
   49. Create a bool variable named lastHitBool
   50. Create an integer type array named winsArray make it have 7 rows and input 0 into each row

# Gameplay cpp:

1. Include Gameplay header

## Starter Function

1. Create starter function
   1. Create integer variable named joeSuccessrate and set it to 50
   2. Create integer variable named sidSuccessrate and set it to 50
   3. Create integer variables named joe and sid
   4. Create bool type variable named run and set it to true
   5. Create while loop and make it loop if run is true
      1. Create integer variable named x and set it a random number between 1-100
      2. Create integer variable named y and setit a random number between 1-100
      3. Create integer variable named I and set it a random number between 0-21
      4. Create integer variable named j and set it a random number between 0-21
      5. If x is less than or equal to joeSuccessrate
         1. Add 50 to joe
         2. Print “Joe hit the bullseye”
      6. Else if (getBoard(i) is greater than 0
         1. Add getBoard(i) to joe
         2. Print Joe missed the bullseye however joe got getBoard(i)
      7. Else if getBoard(i) equals 0
         1. Print Joe missed the board
      8. If y is less than or equal to sidSuccessrate
         1. Add 50 to sid
         2. Print “Sid hit the bullseye”
      9. Else if (getBoard(j) is greater than 0
         1. Add getBoard(j) to sid
         2. Print sid missed the bullseye however sid got getBoard(j)
      10. Else if getBoard(j) equals 0
          1. Print sid missed the board
      11. If joe is greater than sid
          1. Print joe will go first
          2. starterName = “Joe”
          3. Set run to false
      12. If sid is greater than joe
          1. Print sid will go first
          2. starterNamer = “sid”
          3. Set run to false
      13. If joe and sid are equal
          1. Print Joe and sid tied they will have to throw again

## Final function

1. Create final function
   1. Create integer x
   2. Print “How many championships would you like to play?
   3. Input into x
   4. Play startup function
   5. For loop int I = 0, I < x, i++
      1. While joes sets are less than 7 and sids sets are less than 7
         1. Play set function
      2. If joes sets are equal to 7
         1. Winner equals joe
         2. Loser equals sid
      3. If sids sets are equal to 7
         1. Winner equals sid
         2. Loser equals joe
      4. Print The winner of the game was (get winners name)
      5. Add 1 to totalFinals using function
      6. Play frequency table function
      7. Set joes sets to 0
      8. Set sids sets to 0

## Set function

1. Create set function
   1. Play startup function
   2. While joes games are less than 3 and sids games are less than 3
      1. Play the game function
   3. If joes games are equal to 3
      1. Winner equals joe
      2. Loser equals sid
   4. If sids games are equal to 3
      1. Winner equals sid
      2. Loser equals joe
   5. Print The winner of the set was (get winners name)
   6. Set joes sets to 0
   7. Set sids sets to 0

## Game function

1. Create game function
   1. While true loop
      1. Play starter function
      2. If starterName equals joes name
         1. Player points to joe
         2. Break
      3. Else if starterName equals sids name
         1. Player points to sid
         2. Break
   2. Bool run = true
   3. While run is true loop
      1. For loop that plays 3 times
         1. Print player name “turn”
         2. Print player name “score:” player score
         3. Set players lastHit to 0
         4. Play aim function
         5. Play aimCase function
         6. If player score equals 0
            1. Play lastHitCheck function
            2. If lastHitBool equals true

Print player name “Wins!!”

Play average function

Use addGames function and add 1

Set joes score to 501

Set sids score to 501

Set run to false

* + 1. Use setLastHit function and set it to 0
    2. If player pointer equals joe
       1. Player points to sid
    3. Else
       1. Player points to joe

## Startup function

1. Create startup function
   1. Set sid objectname to “Sid”
   2. Set joe object name to “Joe”
   3. Print welcome to the darts game
   4. Print choose joes success rate
   5. Input answer into joe successrate
   6. Print choose sids success rate
   7. Input answer into sid successrate

Frequency Table Function

1. Create frequencyTable function
   1. For loop that loops 7 times int I = 0, I < 7, i++
      1. If joes sets equal 7 and sids sets equals i
         1. Add I to joe winsArray using function
      2. If sids sets equal to 7 and joes sets equal i
         1. Add I to sid winsArray using function
   2. Print “Joes Frequency Table”
   3. Print “joe : Sid : Frequency”
   4. Print”7 : 6 :” winsArray(6) / totalFinals \* 100 %
   5. Print”7 : 5 :” winsArray(6) / totalFinals \* 100 %
   6. Print”7 : 4 :” winsArray(6) / totalFinals \* 100 %
   7. Print”7 : 3 :” winsArray(6) / totalFinals \* 100 %
   8. Print”7 : 2 :” winsArray(6) / totalFinals \* 100 %
   9. Print”7 : 1 :” winsArray(6) / totalFinals \* 100 %
   10. Print”7 : 0 :” winsArray(6) / totalFinals \* 100 %
   11. Print “Sids Frequency Table”
   12. Print “Sid : Joe : Frequency”
   13. Print”7 : 6 :” winsArray(6) / totalFinals \* 100 %
   14. Print”7 : 5 :” winsArray(6) / totalFinals \* 100 %
   15. Print”7 : 4 :” winsArray(6) / totalFinals \* 100 %
   16. Print”7 : 3 :” winsArray(6) / totalFinals \* 100 %
   17. Print”7 : 2 :” winsArray(6) / totalFinals \* 100 %
   18. Print”7 : 1 :” winsArray(6) / totalFinals \* 100 %
   19. Print”7 : 0 :” winsArray(6) / totalFinals \* 100 %

## Aim Case Function

1. Create aimCase function with parameter player pointer
   1. This player is equal to the player passed through the function parameters
   2. Switch (get players aimCase using function)
      1. Case 1:
         1. Throw for treble 20 using function
         2. Break
      2. Case 2:
         1. Throw for double 20 using function
         2. Break
      3. Case 3:
         1. Throw for treble using function
         2. Break
      4. Case 4:
         1. Throw for bullseye using function
         2. Break
      5. Case 5:
         1. Throw for double 10 using function
         2. Break
      6. Case 6:
         1. Throw for treble 7 using function
         2. Break
      7. Case 7:
         1. Throw for using function
         2. Break

## Board cpp:

1. Include board header file
2. Create single function with parameters integer aim and player pointer
   1. Have the player pointer created in the board class equal the player pointer passed through the function parameters
   2. Create integer x and set it a number between 1-100
   3. Create integer I and set it a number between 0-21
   4. If aim equals 25
      1. If x is less than or equal to player successrate
         1. Subtract players score by aim using function
         2. Print “you hit your target aim” aim “your score is now” get player score using function
         3. Add one to players hits using function
         4. Add one to players totalTurns using function
         5. Set players lastHit to aim using function
         6. Add aim to players lastHitTotal using function
      2. If x is less than or equal to 90
         1. Subtract players score by 50 using function
         2. Print “you hit your target aim” 50 “your score is now” get player score using function
         3. Add 50 to players hits using function
         4. Add one to players totalTurns using function
         5. Set players lastHit to 50 using function
         6. Add 50 to players lastHitTotal using function
      3. Else
         1. Set aim to 20
   5. If aim equals 20
      1. If x is less than or equal to player successrate
         1. Subtract players score by aim using function
         2. Print “you hit your target aim” aim “your score is now” get player score using function
         3. Add one to players hits using function
         4. Add one to players totalTurns using function
         5. Set players lastHit to aim using function
         6. Add aim to players lastHitTotal using function
      2. Else if x is less than 92
         1. Subtract players score by ring[0][aim] using function
         2. Print “you hit to the left of your target” ring[0][aim] “your score is now” get player score using function
         3. Add one to players totalTurns using function
         4. Set players lastHit to ring[0][aim] using function
         5. Add ring[0][aim] to players lastHitTotal using function
      3. Else if x is less than 96
         1. Subtract players score by ring[1][aim] using function
         2. Print “you hit to the right of your target” ring[1][aim] “your score is now” get player score using function
         3. Add one to players totalTurns using function
         4. Set players lastHit to ring[1][aim] using function
         5. Add ring[1][aim] to players lastHitTotal using function
      4. If x is less than 98
         1. Subtract players score by 3\* aim using function
         2. Print “you hit a triple:” 3\*aim “your score is now” get player score using function
         3. Add one to players totalTurns using function
         4. Set players lastHit to 3\*aim using function
         5. Add 3\*aim to players lastHitTotal using function
      5. Else
         1. Subtract players score by 2\* aim using function
         2. Print “you hit a double:” 2\*aim “your score is now” get player score using function
         3. Add one to players totalTurns using function
         4. Set players lastHit to 2\*aim using function
         5. Add 2\*aim to players lastHitTotal using function

## Double Function:

1. Create double function with parameters integer aim and player pointer
   1. Have the player pointer created in the board class equal the player pointer passed through the function parameters
   2. Create integer x and set it a number between 1-100
   3. Create integer I and set it a number between 0-21
   4. If x is less than player successrate
      1. Subtract players score by 2\*aim using function
      2. Print “you hit your target aim” 2\*aim “your score is now” get player score using function
      3. Add one to players hits using function
      4. Add one to players totalTurns using function
      5. Set players lastHit to 2\*aim using function
      6. Add 2\*aim to players lastHitTotal using function
   5. Else if x is less than 85
      1. Print “you missed the board”
      2. Add 1 to players total turns using function
      3. Set players lastHit to 0 using function
   6. Else if x is less than 90
      1. Subtract player score by aim using function
      2. Print “you hit below your target” aim “your score is now” get player score using function
      3. Add one to players total turns using function
      4. Set players last hit to aim using function
      5. Add aim to players lastHitTotal using function
   7. Else if x is less than 93
      1. Subtract player score by 2\* ring[0][aim] using function
      2. Add 1 to players totalTurns using function
      3. Print “you hit to the left of your target” 2\* ring[0][aim] “your score is” get player score using function
      4. Set players lastHit to 2\* ring[0][aim] using function
      5. Add 2\* ring[0][aim] to players lastHitTotal using function
   8. Else if x is less than 96
      1. Subtract players score by 2\* ring[1][aim] using function
      2. Add one to players totalTurns using function
      3. Print “you hit to the right of your target” 2\* ring[1][aim] “your score is now” get player score using function
      4. Set players last hit to 2\* ring[1][aim] using function
      5. Add 2\* ring[1][aim] to players lastHitTotal using function
   9. Else if x is less than 98
      1. Subtract player score by ring[0][aim] using function
      2. Add one to players totalTurns using function
      3. Print “you hit to the left of the target” ring[0][aim] “your score is now” get player score using function
      4. Set players lastHit to ring[0][aim] using function
      5. Add ring[0][aim] to players lastHitTotal using function
   10. Else
       1. Subtract players score by ring[1][aim] using function
       2. Add one to players totalTurns using function
       3. Print “You hit to the right of your target” ring[1][aim] “your score is now “ get players score using function
       4. Set player lastHit to ring[1][aim] using function
       5. Add ring[1][aim] to players lastHitTotal using function

## Treble Function:

1. Create treble function with parameters integer aim and player pointer
   1. Have the player pointer created in the board class equal the player pointer passed through the function parameters
   2. Create integer x and set it a number between 1-100
   3. Create integer I and set it a number between 0-21
   4. If x is less than player successrate
      1. Subtract players score by 3\*aim using function
      2. Print “you hit your target aim” 3\*aim “your score is now” get player score using function
      3. Add one to players hits using function
      4. Add one to players totalTurns using function
      5. Set players lastHit to 3\*aim using function
      6. Add 3\*aim to players lastHitTotal using function
   5. Else if x is less than 85
      1. Print “you missed the board”
      2. Add 1 to players total turns using function
      3. Set players lastHit to 0 using function
   6. Else if x is less than 90
      1. Subtract player score by aim using function
      2. Print “you hit below your target” aim “your score is now” get player score using function
      3. Add one to players total turns using function
      4. Set players last hit to aim using function
      5. Add aim to players lastHitTotal using function
   7. Else if x is less than 93
      1. Subtract player score by 3\* ring[0][aim] using function
      2. Add 1 to players totalTurns using function
      3. Print “you hit to the left of your target” 3\* ring[0][aim] “your score is” get player score using function
      4. Set players lastHit to 3\* ring[0][aim] using function
      5. Add 3\* ring[0][aim] to players lastHitTotal using function
   8. Else if x is less than 96
      1. Subtract players score by 3\* ring[1][aim] using function
      2. Add one to players totalTurns using function
      3. Print “you hit to the right of your target” 3\* ring[1][aim] “your score is now” get player score using function
      4. Set players last hit to 3\* ring[1][aim] using function
      5. Add 3\* ring[1][aim] to players lastHitTotal using function
   9. Else if x is less than 98
      1. Subtract player score by ring[0][aim] using function
      2. Add one to players totalTurns using function
      3. Print “you hit to the left of the target” ring[0][aim] “your score is now” get player score using function
      4. Set players lastHit to ring[0][aim] using function
      5. Add ring[0][aim] to players lastHitTotal using function
   10. Else
       1. Subtract players score by ring[1][aim] using function
       2. Add one to players totalTurns using function
       3. Print “You hit to the right of your target” ring[1][aim] “your score is now “ get players score using function
       4. Set player lastHit to ring[1][aim] using function
       5. Add ring[1][aim] to players lastHitTotal using function

## Bullseye Function:

1. Create bullseye function
   1. Create integer x and set it a number between 1-100
   2. Create integer I and set it a number between 0-21
   3. If x is less than player successrate
      1. Subtract players score by 50 using function
      2. Print “Bullseye, you got 50 points. Your score is now:” get player score using function
      3. Add one to players hits using function
      4. Add one to totalBullseyes using function
      5. Add one to players totalTurns using function
      6. Set players lastHit to 50 using function
      7. Add 50 to players lastHitTotal using function
   4. Else if board[i] is greater than 0
      1. Subtract player score by board[i] using function
      2. Print “You missed the bullsey However you got” board[i] “ your score is now” get player score using function
      3. Add one to players totalTurns using function
      4. Set players lastHit to board[i] using function
      5. Add board[i] to players lastHitTotal using function
   5. Else if board[i] equals 0
      1. Print “you missed the board”
      2. Add one to players totalTurns using function
      3. Set players lastHit to 0 using function

# Player cpp

1. Include player header

## Average function

1. Create average function
   1. Player average equals hits divided by totalTurns \* 100
   2. Print “Player average:” playerAverage “%”
   3. Print bullseyes hit “ totalBullseyes

## Aim function

1. Create aim function
   1. If score is greater than 60
      1. aimCase equals 1
   2. else if score is even and score is lessthan or equal to 60 and greater than or equal to 51
      1. aimCase equals 2
   3. else if score is odd and score is lessthan or equal to 60 and greater than or equal to 51
      1. aimCase equals 1
   4. else if score equals 50
      1. aimCase equals 3
   5. else if score is even and score is lessthan or equal to 49 and greater than or equal to 41
      1. aimCase equals 4
   6. else if score is odd and score is lessthan or equal to 49 and greater than or equal to 41
      1. aimCase equals 5
   7. else if score is even and score is lessthan or equal to 40 and greater than or equal to 20
      1. aimCase equals 6
   8. else if score is even and score is lessthan or equal to 20
      1. aimCase equals 6
   9. else if score is odd
      1. aimCase equals 7

## lastHitCheck function

1. create lastHitCheck function
   1. integer x equals getLastHit
   2. if x is even and x is less than or equal to 40 or x equals 50
      1. lastHitBool equals true
   3. else
      1. add lastHitTotal to score using function
      2. print “your score has been reset too:” score
      3. lastHitBool equals false
   4. set lastHit to 0 using function
   5. set lastHitTotal to 0 using function